Pascal NEWSLETTER

PARTY EDITION OREGON SOFTWARE MAY 1982

5th annual open house set for May 14th

At a recent Friday night get-together at Jake's Famous Crayfish House, Oregon Software announced its 5th annual Open House to a large group of uninterested parties.

The event, according to company President Rusty Whitney, will take place May 14, 1982, beginning at 2 p.m. Under development for one year, the new open house promises to exceed all of the competition's party products. In describing the reasons for developing this latest version, Whitney said, "Well, nobody could really remember last year's party, so we decided to do it again."

New features include previously non-existent staff, potentially future non-existent staff, and guided tours of Oregon Software's newly acquired and as yet unremodeled dungeon. Whitney also emphasized that the best features of previous open houses will be retained: imported beer and gluttonous quantities of good food.

As a promotional gimmick, this unsupported product will be offered free to anyone who can find Oregon Software's World Headquarters (the word from Oregon Software's distributors in Sweden, France, England, Japan, Holland, and Australia is that each has already fielded a team).

Water-cooled Pascal offered

In an effort to remain a leader in the field of Pascal, Oregon Software's engineering department began work today on a compiler for the Univac I. Bud Snocca, staff programmer, described the decision process as "a difficult one. We didn't feel it was fair that simply because the machine was designed and manufactured before the era of Pascal, it should never have the chance to join in the software revolution. It's sort of like the trilobites never having a chance to be saved from extinction."

Development will be hindered by the fact that a functional Univac I is not available. However, this alleviates the debugging problem. In addition, customer support should be a breeze. (For sites located north of the Tropic of Cancer, one gallon of anti-freeze will be included with each order.)

Letters

Dear Sir:

Last week I received your Pascal compiler for my DEC PDP-11/23 processor. I had ordered it against the advice of both my butcher and neighborhood dog catcher (but I figured, what do they know anyway?) They agreed with my mother-in-law that Oregon Software sounded perverted. Well, I had no sooner installed it on my system than all hell broke loose. The machine (which had functioned perfectly up to this time) exploded into 4,637 pieces spread across a 12-foot radius. Let me tell you...it's difficult to impress my friends and neighbors with 4,637 pieces of junk!

Strugg Ling Hacker

PhD finds Pascal benchmark

Jean Pierre Hors D'Oeuvres, noted French archaeologist from the Universite du Merde du Tareau, shocked the computer science community last week when he announced his discovery of the genuine Pascal Benchmark. Repercussions were felt around the world as computer science departments rushed to destroy all evidence of their imitation benchmarks. Advertising agencies immediately issued disclaimers to computer industry journals.

Dr. Hors D'Oeuvres had devoted 3 long years toward finding his prize. "I carefully combed the grounds around Pascal's former residences, searching for the Benchmark," he said. "I was sure I would find it. Suddenly, I fell over a tiny projection in what was formerly a vegetable garden. It was next to a brussels sprout. I madly began to dig with my fingernails. I knew I had found it at last!"

Arthur Whales, from the University of Plasmania, has challenged Dr. Hors D'Oeuvres to validate his find. How 'suite' it is!

Inside this issue...

5th Annual Open House Announced . . . . . . Page 1
Water-Cooled Pascal Offered . . . . Page 1
Archaeologist finds Pascal benchmark . . . . . Page 1
Pascal and Ada 'seek' Divorce . . . . Page 2
Post-Mortem Dump Causes Scatological Furor . Page 6
Micro-Puny Pascal Critiqued . . . . Page 8

Oregon Software
2340 SW Canyon Road
Portland, OR 97201

PORTLAND, OR 97208
PM
21 APR
1982

PORTLAND
APR 21 82
OREG.
U.S. POSTAGE
.20
B METER
022167

ALWAYS USE
ZIP CODE

John Peterson
1128 Austin Hall
Univ. of Utah
Salt Lake City, UTAH
84112

OPEN HOUSE

Pascal-2™

Software Product Description #SPD-1135

- Advanced-design Pascal Compiler with Debugger, Profiler, and Utilities.
- For all Digital PDP-11 processors, including the LSI-11 and the VAX-11 in compatibility mode.
- For all Digital PDP-11 operating systems:
  - RSTS/E (V6C and V7)
  - RSX-11 (11M, 11D, 11M-Plus, IAS, VMS/AME)
  - RT-11 (V3, V4, SJ, XM, TSX-Plus)
- Includes two copies of **Pascal–2 User Manual** with User's and Programmer's Guides, Language Specification, Debugger, Utilities, Conversion, and Installation Guides, plus **Programming in Pascal** and **Pascal User Manual and Report** textbooks.
- Includes **Software Support** for one year from purchase (renewable):
  - Telephone consultation
  - 30-day written response to **Trouble Reports**
  - **Maintenance Releases** at no extra charge
  - Oregon Software's **Pascal Newsletter**
- Compiler requires EIS extended instruction set (standard on new PDP-11 processors), 28K words of memory, and 500 blocks of disk storage. Not recommended for compilation on single-density floppy-based systems. Generated code does not require EIS and supports EIS, FIS, and FPP hardware if available.
- Media available: 9-track magtape, 800 and 1600 bpi; floppy disk, single and double density; RK05 and RL01 cartridge disks. Write for price quotes on other media.
- **Binary Licenses** for one operating system:
  - CPU (single processor) — $3450
  - Site (multiple processors, one location) — $3950
  - Instructional not-for-profit site — $1450
  - Additional operating system — $1450
  - Campus or Facility (multiple processors, contiguous buildings) — $11,700
  - Corporate-wide licenses — Write for price quote
- **Source Licenses** (a Binary License is a prerequsite):
  - Run-Time Library, source code in MACRO-11 — $700
  - Debugger, source code in Pascal — $2000 (requires a separate license)
  - Compiler, source code in Pascal — $12,600 (requires a separate license)
- Prices are in US dollars and include shipment via UPS and media charges.
- Shipment within 20 business days after receipt of purchase order and completed licenses; shipment of prepaid orders within 3 business days.
- Money-back warranty: If, within 30 days of receipt, you determine that **Pascal–2** does not conform to this Product Description, you may return the software for a full refund.

PDP, VAX, RSTS/E, RSX, and IAS are trademarks of Digital Equipment Corporation. TSX-Plus is a trademark of S & H Computing, Inc. **Pascal-1** and **Pascal-2** are trademarks of Oregon Software.

Printed in USA May 15, 1981. © 1981 Oregon Software. All Rights Reserved.

Oregon Software 2340 S.W. Canyon Road • Portland, Oregon 97201 (503) 226-7760 TWX: 910-464-4779

Pascal-2 Compiler Technical Description

Performance

The **Pascal–2** compiler, an optimizing compiler of advanced design, produces object code with performance that is unsurpassed by any compiler previously available on the PDP-11. The code produced by **Pascal–2** compares favorably in size and speed with that produced by Digital's FORTRAN IV Plus compiler and is both smaller and an order of magnitude faster than interpretive or threaded languages (BASIC-Plus, BASIC-Plus-2, UCSD Pascal). **Pascal–2** code is typically 30% to 40% smaller and twice as fast as code produced by our **Pascal–1** compiler.

Operating in five phases, the compiler employs virtual memory techniques to allow compilation of very large programs. Optimizations include global register allocation, constant folding, dead code elimination, short-circuit evaluation, expression targeting, array index simplification, branch-tail merging, and common subexpression elimination.

Compilation speed is dependent on processor type, system loading, and disk configuration. On a lightly loaded RSTS/E PDP-11/44 with an RP06 disk, large programs typically compile at speeds of 800 to 1400 lines per minute. On a single-user RT-11 XM PDP-11/23 with RX02 floppy disks, large programs typically compile at 160 to 240 lines per minute.

Language

The language accepted by the **Pascal–2** compiler conforms closely to the draft proposed Pascal standard (International Standards Organization dp7185.1). ISO conformance is Level 0: Conformant array parameters are not included. All capabilities of standard Pascal are supported in full. These capabilities include procedure and function parameters, non-local goto statements, set types of up to 256 elements, and packed data structures. **Pascal–2** supports integer types from 1 to 16 bits and single-precision or double-precision real types. Identifiers may be of any length, with all characters significant.

The **Pascal–2** language includes a number of extensions for programmer convenience, I/O flexibility, and low-level operations. Convenience extensions include: "external" procedures, which may be **Pascal–1** or **Pascal–2**; "nonpascal" external calls using the Digital standard calling sequence (languages such as FORTRAN or MACRO-11); an "include" directive, for combining multiple source files in a compilation; flexible ordering of declaration sections; structured constant declarations; and a default "otherwise" clause in the "case" statement. I/O extensions include random-access files and additional parameters in "reset" and "rewrite" to specify external filenames and I/O mode options. Low-level extensions include: "and", "or", and "not" operators on integer types; "origin" variable declaration that assigns a variable to a specific address, such as a UNIBUS device register; octal constants; "ref" pointer generation function; "loophole" function for type coercion; and "size" and "bitsize" memory allocation functions.

A standard compilation switch will disable all of the compiler's extended language features, allowing only elements of the ISO standard language to be compiled.

Error Checking

Type compatibility checks ensure that data types strictly conform to the ISO standard; other checks detect many uninitialized variables and other errors normally found only during execution. Error messages describe (in succinct English) the kind of error and display the source line in error and the position of the error. The compiler is not prone to "ripple" or cascade effects from previous errors.

All error checking is done in the first two phases of the compilation process; detection of errors causes later phases to be bypassed.

Run-time checking detects array index errors, subrange assignment errors, nil pointer references, non-existent case labels, and many I/O and arithmetic errors. (Run-time checking may be disabled for maximum performance.)

Operation

The **Pascal–2** compiler accepts one or more source files and directly produces relocatable binary object files compatible with Digital's Linker, Task Builder, and Librarian. A compilation option will generate assembler source code that can be assembled by standard assemblers. The source input to the compiler may be a complete program or a module composed of procedures and functions to be compiled separately and later linked with other routines.

Regardless of the host machine, the **Pascal–2** compiler can generate code for processors with any (or none) of the EIS, FIS, or FPP hardware options.

The compiler requires 28K words of main memory, approximately 500 blocks of disk storage, and the EIS hardware option. The RT-11 FB (foreground/background) monitor does not provide sufficient memory for compiler operation; programs may be compiled under the SJ or XM monitors and then run under the FB monitor. The compiler will operate on a system based on dual single-density floppy disks, but we recommend larger disks for acceptable performance.

Reliability

Written in Pascal and designed with structured methods, the **Pascal–2** compiler includes internal overflow and consistency checks. Extensively field-tested since May 1980, **Pascal–2** has proven to be stable and robust in diverse user environments.

Pascal-2 Debugger and Profiler

The **Pascal–2 Debugger** gives the programmer complete interactive control of program execution in a high-level Pascal context. The **Debugger** is conversant with all standard and programmer-defined identifiers, constants, types, and variables. The **Debugger** provides access to all of the Pascal data structures (including records, arrays, pointers, sets, and files, but excluding structured constants) in the same notation used in the program.

Debugger control capabilities include: full trace; statement-by-statement execution, allowing the execution of an entire procedure as a single statement or the detailed execution of all procedures; and any number of control breakpoints identified by block name and statement number, with automatic execution of stored commands. Using the interrupt feature (Control-C), a programmer can break into program execution at any time: no advance action is required. In addition, run-time errors transfer control to the **Debugger** to allow immediate identification of the problem.

Each breakpoint is identified by display of the block name, statement number, and the Pascal source code from the program. The **Debugger** makes the entire program listing available to the programmer, who may interactively display the current statement, current procedure, or any other portion of the source program. At any time, a history of program execution can show the previous 50 statements executed along with the current stack frame structure, including all blocks pending execution.

Debugger data commands permit display and/or modification, using Pascal notation, of all existing variables. The **Debugger** includes convenience capabilities not available in Pascal programs, such as the ability to write a complex array or record structure with one statement. All variables are accessible to the **Debugger**, including those in pending stack frames.

Another **Debugger** command allows the programmer to "watch" any number of variables (or absolute memory locations) and to interrupt the program at any statement that modifies those variables.

The compilation switch that invokes the **Debugger** causes the generation of embedded control instructions. The **Debugger** overlays the program being debugged and adds only 2K to 6K words of overhead; thus, the **Debugger** can be used with programs close to the limits of memory.

Pascal–2 Profiler

The Profiler is a measurement tool that helps identify program bottlenecks and points out those sections of the program where a small amount of programmer effort may have large payoffs in execution efficiency. During program execution, the Profiler counts each statement executed within every block of the program. When the program finishes, the Profiler displays a modified listing of the program with the actual execution count attached to each source line. The Profiler also summarizes each procedure and function and indicates the percent of the execution count taken by each program block.

Pascal–2 Utility Programs

The **Pascal–2** system comes with utility programs and packages to aid the applications programmer with development, maintenance, and documentation. Except for PASMAC, the utilities are written in **Pascal–2**; all are supplied in source form.

- PASMAT, a source formatter for **Pascal–2** programs, automatically indents and formats programs to ensure a consistent and readable structure. PASMAT includes many options to allow the programmer control over the formatting process.
- PB, another **Pascal–2** program formatter, does only the tedious work of consistent indention and capitalization. PB is compatible with a wide variety of individual formatting tastes.
- XREF, a cross-reference index generator, displays each identifier and all of its uses. XREF will also identify definition points and "assigning references" to variables.
- PROCREF, a procedural cross-reference index program, displays the procedure and function nesting structure and identifies "calls/called by" for each block of the program, including external references.
- STRING, written in standard Pascal, is a collection of procedures and functions for programming with dynamic character strings. Programs using STRING can be transported to any Pascal implementation.
- PASMAC, a library of MACRO-11 definitions, aids the interfacing of **Pascal–2** programs and assembler routines. PASMAC provides macro definitions for entry and exit points, parameter and variable stack references, register save/restore facilities, and simple data structure definitions.
- PROSE, a documentation aid similar to RUNOFF, provides facilities for automatic paragraphing and layout of text. PROSE can produce finished documents with headers, footnotes, justified margins, and indices or content tables.

Oregon
Software
2340 S.W. Canyon Road
Portland, Oregon 97201
(503) 226-7760
TWX: 910-464-4779

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. A-407 PORTLAND, OREGON

POSTAGE WILL BE PAID BY ADDRESSEE

Oregon Software

2340 S.W. Canyon Road
Portland, Oregon 97201

Pascal-2™

High-Performance Structured Programming

Send information on **Pascal-2**™

☐ *Why Pascal?* brochure

☐ **Pascal-2**™ User Manual

☐ RT-11 ☐ RSX/IAS ☐ RSTS/E

☐ Product Description

☐ Benchmark Study

☐ Sample Pascal programs

☐ Newsletter sample copy

☐ Have a representative telephone me

I am specifically interested in:

____

Name ______________________________

Company ______________________________

Address ______________________________

City ______________ State ______________ Zip ______________

Telephone (   ) ________ - ______________ x ________

Oregon
Software

HISTORY OF OREGON SOFTWARE (ONCE CALLED "OMSI")

Oregon Software traces its origins to the Oregon Museum of Science and Industry in Portland (OMSI), a private, non-profit educational organization. Chartered "to enhance the general public understanding of science and technology," OMSI operated a research laboratory in the 1970s that, along with experiences in the physical and biological sciences, provided the means for students to learn about computers and programming. One ground rule established by Rusty Whitney, director of the lab, was that students had to produce programs useful for other people. They became so good at this that many OMSI programs spread around the world through informal networks of users. OMSI programmers whose work became widely used included Don Baccus (OMSI BASIC), Barry Smith (OMSI FOCAL) and Steve Poulsen (OMSI APL). Early on, the word "OMSI" became associated with quality programming.

In September of 1977, seven people, with Rusty as president, came from OMSI to found Oregon Software. The goal was to explore the technical and commercial potential of a new structured programming language, Pascal. The name "OMSI" stayed for a while as an acronym for Oregon Minicomputer Software, Inc. Even though we simplified our name to Oregon Software, many customers still call us OMSI, and two of our early products, Pascal-1 and Sort-1-Plus, remain better known as OMSI Pascal and OMSI Sort.

For all that, we're not OMSI the science museum -- though we continue to support OMSI and its educational programs. Oregon Software is a research and development group in Portland that creates system development software. We have a staff of 35, a chain of 25 distributors worldwide, and roughly 3,000 users of our products. We expect to have about $2.5 million in sales this fiscal year and $3.8 million in FY 1983. Our primary product is a high-performance Pascal compiler and debugger, with an assortment of utilities for professional programmers. That package is Pascal-2.

2340 SW Canyon Road, Portland, Oregon 97201
503/226-7760 TWX 910-464-4779

History of Pascal-2

Our experience with Pascal-1 made us realize that many of our best ideas --- whether for code generation or error reporting -- wouldn't work in our single-pass compiler. In addition, Pascal-1 had a number of low-level extensions that aided programming on the PDP-11 but limited portability to other machines. We began, therefore, to develop a multi-pass compiler that emphasized conformance to the Pascal standard (and, hence, portability) while generating optimized code. We were determined that this compiler would be the most efficient we knew how to make, partly because we wanted the best possible tool for our own use.

It took us five years, but we believe we succeeded. Pascal-2 combines the virtues of a structured, high-level language with the performance of a low-level language, as borne out by benchmark tests in the accompanying chart. The compiler itself is portable, having been installed on three computers made by different manufacturers. Our native Pascal-2 compilers for the PDP-11 (running on RSX, RSTS/E, RT-11, and TSX-Plus) have been on the market for more than a year.

A Look to New Markets

Having established ourselves as an end-user supplier with a reputation for quality and reliability, we're now determined to expand our entry into the OEM market for both the PDP-11 and the Motorola MC68000. We're now offering OEMs a choice of two system development packages for the Motorola MC68000. The cross-compiler package consists of our RSX compiler and utilities, a cross-compiler from RSX to the VERSAdos operating system on the MC68000, and a data transfer utility. The native package consists of a native compiler for VERSAdos and the standard Pascal-2 utility package, including Debugger and Profiler. A stand-alone package for the MC68000 is available with either package, providing concurrent programming capability. (See the accompanying product descriptions for details.) These are being made available to OEMs before end users.

Other products for DEC machines include a UNIX compiler (in field test) and a VAX/VMS native compiler (under development). Other plans for the MC68000 include a MC68000-based UNIX product and other combinations of cross-compilers with the MC68000 as the target (these are being negotiated with OEMs). As the pioneer in performance Pascal, we want to make Pascal-2 the standard language for system development on the MC68000 and the PDP-11.

Preliminary Software Product Description
OEM System Development Packages for the MC68000
Oregon Software

This document describes Oregon Software's Pascal-2 system development packages for the Motorola MC68000. The products are as follows:

-- The RSX cross-compiler package: an advanced-design Pascal native compiler and a cross-compiler running on RSX-11 operating systems (11M, 11M-Plus, VMS/AME), including the the VAX-11 in compatibility mode, for the Digital PDP-11 processor; a Debugger, Profiler, and other utilities for RSX. The cross-compiler generates object files that can be linked and run under the VERSAdos operating system on the MC68000 and includes XFR, a utility that can transfer object files or text files from RSX to VERSAdos and back.

-- The VERSAdos native package: the Pascal-2 compiler processing the same language as the cross-compiler; a Debugger, Profiler, and other utilities running on VERSAdos, described in detail below.

-- An MC68000 package providing concurrent programming in standard Pascal and allowing programs to run without an operating system on the MC68000; available as an option for either the RSX cross-compiler or the VERSAdos compiler.

Details of the RSX cross-compiler and the VERSAdos native compiler are described in this document, along with the utilities accompanying the VERSAdos compiler. Also described is the XFR utility, available only with the RSX cross-compiler.

Details of the RSX native compiler and utilities are outlined in the accompanying Product Description, 1135. Details of the stand-alone package for the MC68000 are described in the accompanying Product Description.

The RSX native package, the RSX cross-compiler, and the VERSAdos native package may be purchased separately. The stand-alone system, which consists of library routines that enable generated code to run without an operating system, can be purchased only as an option to one of the compilers.

Manual and Report; sections on cross-compiling and concurrent programming are included.

Media: The cross-compiler is available on 9-track magtape, 800 and 1600 bpi; floppy disk, single and double density; and RK05, RL02, and RL01 cartridge disks. Write for price quotes on other media. The native compiler is initially available only on floppy disk.

Shipment within 20 business days after receipt of purchase order and completed licenses; shipment of prepaid orders within 3 business days.

PDP, RSX, and VAX are trademarks of Digital Equipment Corporation. MC68000 and VERSAdos are trademarks of Motorola, Inc. Pascal-2 is a trademark of Oregon Software, Inc.

Printed in USA July 1982
Copyright 1982 Oregon Software. All Rights Reserved.

Pascal-2 Compiler Technical Description

This section describes the VERSAdos native compiler and the RSX cross-compiler, both of which generate code that runs under the VERSAdos operating system.

Performance

Operating in five phases, the compiler employs virtual memory techniques to allow compilation of very large programs. Optimizations include global register allocation, constant folding, dead code elimination, short-circuit evaluation, expression targeting, array index simplification, branch-tail merging, and common subexpression elimination.

Language

The language accepted by the Pascal-2 compiler conforms closely to the draft proposed Pascal standard (International Standards Organization dp7185.1). ISO conformance is Level 0: Conformant array parameters are not included. All capabilities of standard Pascal are supported in full. These capabilities include procedure and function parameters, non-local goto statements, set types of up to 256 elements, and packed data structures. Pascal-2 supports integer types from 1 to 32 bits (16 bits in the field-test version) and single-precision or double-precision real types. Identifiers may be of any length, with all characters significant.

The Pascal-2 language includes a number of extensions for programmer convenience, I/O flexibility, and low-level operations. Convenience extensions include: "external" procedures, for separate compilation; an "include" directive, for combining multiple source files in one compilation; flexible ordering of declaration sections; structured constant declarations; and a default "otherwise" clause in the "case" statement. I/O extensions include random-access files and additional parameters in "reset" and "rewrite" to specify external file names and I/O mode options. Low-level extensions include: "and", "or", and "not" operators on integer types; "origin" variable declaration that assigns a variable to a specific address; octal and hexidecimal constants; "ref" pointer generation function; "loophole" function for type coercion; and "size" and "bitsize" memory allocation functions.

A standard compilation switch will disable all of the compiler´s extended language features, allowing only elements of the ISO standard language to be compiled.

Error Checking

Type compatibility checks ensure that data types strictly conform to the ISO standard; other checks detect many uninitialized variables and other errors normally found only during execution. Error messages describe (in succinct English) the kind of error and display the source line in error and the position of the error. The compiler is not prone to "ripple" or cascade effects from previous errors.

All error checking is done in the first two phases of the compilation process; detection of errors causes later phases to be bypassed.

Run-time checking detects array index errors, subrange assignment errors, invalid pointer references, non-existent case labels, stack or heap overflow, and many I/O and arithmetic errors. An error "walkback" feature will automatically print the execution history of the procedures that lead to an error. (Run-time checking may be disabled for maximum performance.)

Operation

The Pascal-2 compiler accepts one or more source files to directly produce relocatable binary object files compatible with the MC68000 Linker. A compilation option will generate assembler source code that can be assembled by the Motorola assembler. The source input to the compiler may be a complete program or a module composed of procedures and functions to be compiled separately and later linked with other routines.

Utilities for the MC68000

This section describes the utilities included with the VERAdos native compiler. Except for a data transfer utility, the cross-compiler does not include utilities. Comparable utilities are included with the RSX native package.

Pascal-2 Debugger and Profiler

The Pascal-2 Debugger gives the programmer complete interactive control of program execution in a high-level Pascal context. The Debugger is conversant with all standard and programmer-defined identifiers, constants, types, and variables. The Debugger provides access to all of the Pascal data structures (including records, arrays, pointers, sets, and files, but excluding structured constants) in the same notation used in the program.

Debugger control capabilities include: full trace; statement-by-statement execution, allowing the execution of an entire procedure as a single statement or the detailed execution of all procedures; and any number of control breakpoints identified by block name and statement number, with automatic execution of stored commands. Using the interrupt feature

(Control-C), a programmer can break into program execution at any time: No advance action is required. In addition, run-time errors transfer control to the Debugger to allow immediate identification of the problem.

Each breakpoint is identified by display of the block name, statement number, and the Pascal source code from the program. The Debugger makes the entire program listing available to the programmer, who may interactively display the current statement, current procedure, or any other portion of the source program. At any time, a history of program execution can show the previous 50 statements executed along with the current stack frame structure, including all blocks pending execution.

Debugger data commands permit display and/or modification, using Pascal notation, of all existing variables. The Debugger includes convenience capabilities not available in Pascal programs, such as the ability to write a complex array or record structure with one statement. All variables are accessible to the Debugger, including those in pending stack frames.

Another Debugger command allows the programmer to "watch" any number of variables (or absolute memory locations) and to interrupt the program at any statement that modifies those variables.

Pascal--2 Profiler

The Profiler is a measurement tool that helps identify program bottlenecks and points out those sections of the program where a small amount of programmer effort may have large payoffs in execution efficiency. During program execution, the Profiler counts each statement executed within every block of the program. When the program finishes, the Profiler displays a modified listing of the program with the actual execution count attached to each source line. The Profiler also summarizes each procedure and function and indicates the percent of the execution count taken by each program block.

Pascal--2 Utility Programs

The Pascal-2 system comes with utility programs and packages to aid the applications programmer with development, maintenance, and documentation. Except for PASMAC, the utilities are written in Pascal-2; all are supplied in source form.

-- PASMAT, a source formatter for Pascal-2 programs, automatically indents and formats programs to ensure a consistent and readable structure. PASMAT includes many options to allow the programmer control over the formatting process.

-- PB, another Pascal-2 program formatter, does only the tedious work of consistent indention and capitalization. PB is compatible with a wide variety of individual formatting tastes.

-- XREF, a cross-reference index generator, displays each identifier and all of its uses. XREF will also identify definition points and "assigning references" to variables.

-- PROCREF, a procedural cross-reference index program, displays the procedure and function nesting structure and identifies "calls/called by" for each block of the program, including external references.

-- STRING, written in standard Pascal, is a collection of procedures and functions for programming with dynamic character strings. Programs using STRING can be transported to any Pascal implementation.

-- PASMAC, a library of assembler definitions, aids the interfacing of Pascal-2 programs and assembler routines. PASMAC provides macro definitions for entry and exit points, parameter and variable stack references, register save/restore facilities, and simple data structure definitions.

-- XFR, a data transfer utility, can transfer object files or text files from the RSX system to the VERSAdos system and back. XFR uses a "hand-shaking" format that guarantees accurate receipt of data and can produce a log on RSX of program execution and other steps on VERSAdos. XFR is available only with the cross-compiler.

Distributor List

INTERNATIONAL

Australia
Network Computer Services
6 Cunningham Street
Sydney 2000
Telephone: 211-2322
TLX: 23872

Austria
Orga-Soft GmbH
Alter Platz 4
A-9020 Klagenfurt
Telephone: 04222/70715
TLX: 42466

Canada
Prior Data Sciences Ltd.
16 Credit Union Way
Suite 301
Nepean, Ontario K2H 8R6
Telephone: (613) 820-7235
TLX: 053-3356

Valley Software, Inc.
6400 Roberts Street, #390
Burnaby, BC V5G 4G2
Telephone: 604-291-0651

Denmark
IMDATA
Smedeland 8
2600 Glostrup
Telephone: 02-45-07-97
TLX: 33285

England
Hourds Computing, Ltd.
7-8 Mill Street
Stafford ST16 2AJ
Telephone: 0785-44221
TLX: 36540

Real Time Products
1 Paul Street
London EC2A 4JJ
Telephone: 01-588-0667
TLX: 884971

Finland
Mimarobe Oy
P.O.B. 361
SF-33101 Tampere 10
Telephone: 358/31 36333
TLX: 22400

France
Yrel Electronique et
Informatique
Rue Fourney Z.1., B.P. 40
78530 Buc
Telephone: 956-81-42
TLX: 696379

Holland
NHCS, B.V.
Kenaupark 33
2011 MR Haarlem
Telephone: 023-324951
TLX: 71289

Software Sciences
Nederland B.V.
A.J. Ernststraat 865
1082 LN Amsterdam
Buitenveldert
Telephone: 020-461331
TLX: 17029

Israel
Triple-D, Ltd.
29, Hayetstra Street,
P.O.B. 3069
Ramat-Gan
Telephone: 03-719195-6
TLX: 341320

Italy
Zeltron
Uffici di Milano
Viale Bianca Maria 45
Milano 20122
Telephone: (02) 709176
TLX: 312099

Japan
Rikei Corporation
1-26-2 Nishi Shinjuku
4316 Shinjuku-ku
Tokyo 160
Telephone: 345-3411
TLX: 24208

Sweden
Sern Computer
Consulting, Ltd.
Karlagatan 1, S-416 61
Gothenburg
Telephone: 031/80-18-20
TLX: 27711

Switzerland
Muehlethaler Electronics AG
Falken 100a
4511 Rumisberg
Telephone: 065/76-26-53
TLX: 34506

West Germany
AC Copy
Kurbrunnenstrasse 30
D-5100 Aachen
Telephone: 0241-506096
TLX: 832368

Periphere Computer
Systems GmbH
Pfalzer-Wald-Strasse 36
D-8000 Munchen 90
Telephone: 089/681021
TLX: 523271

UNITED STATES

California
Software Supply
4618 E. Sixth Street
Long Beach, CA 90814
(213) 434-3723

Theta Business Systems
1110 Sonora Avenue, #106
Glendale, CA 91201
(213) 242-7981

Illinois
Processing Concepts, Ltd.
2909 N. Sheridan Road
Chicago, IL 60657
(312) 883-1444

Maryland
General Scientific Corporation
P.O. Box 2145
Montgomery Village Station
Gaithersburg, MD 20760
(301) 840-2773

Massachusetts
Computer Software
Consultants, Inc.
200 Boylston Street, #407
Chestnut Hill, MA 02167
(617) 964-4316

Structural Programming, Inc.
83 Boston Post Road
Sudbury, MA 01776
(617) 443-5366

Oregon
Interactive Technology, Inc.
1225 N.W. Murray Road, #103
Portland, OR 97229
(503) 644-0111

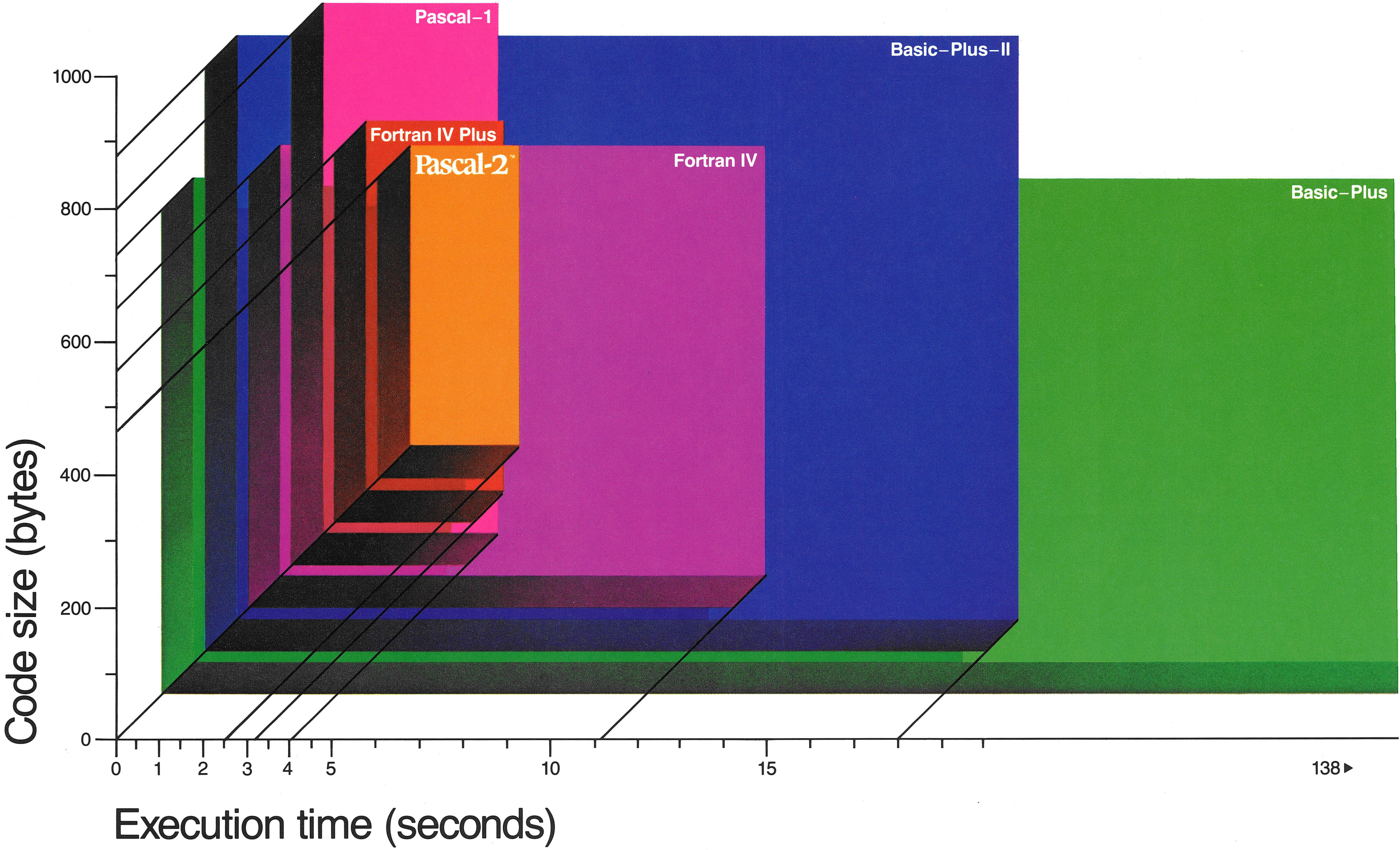
Tennessee
Custom Computer
Applications, Inc.
3876 Delp St.
Memphis, TN 38118
(901) 362-1325

Utah
Terabit Computer Engineering
1314 South 500 East
Salt Lake City, UT 84105
(801) 487-3033

Oregon Software
2340 S.W. Canyon Road
Portland, Oregon 97201
(503) 226-7760
TWX: 910-464-4779

Printed in USA
1-Feb-82

Quicksort Benchmark (PDP-11/45)



Quicksort Benchmark

A Quicksort algorithm described by Robert Sedgewick in *Communications of the Association for Computing Machinery* October 1978, was translated into BASIC, FORTRAN, and Pascal. Execution time and object code size were measured at Oregon Software for several PDP-11 language processors. Object size does not include run-time support routines required by some of the language implementations (BASIC-Plus, FORTRAN IV threaded, Swedish Pascal, and UCSD Pascal).

To the best of our knowledge, non-standard language features were not used. Program listings are available at no charge from Oregon Software. Benchmark data were collected from code executed on a PDP-11/45 RSTS/E system with floating point hardware and on an LSI-11, RT-11 system without floating point hardware, both systems were run without other load.

Times are CPU clock seconds (this benchmark is entirely computational) and object space is in bytes:

	Execution in Seconds		Object Code Size
	PDP-11	LSI-11	in Bytes
Pascal-2	2.5	13	472
FORTRAN IV Plus	3.2	*	558
FORTRAN IV (in-line)	3.3	**	576
NBS Pascal	3.6	*	756
Pascal-1	4.0	24	796
FORTRAN IV (threaded)	11.2	49	648
BASIC Plus-2	16	*	882
Swedish Pascal	16	*	1494
UCSD Pascal	**	280	756
BASIC Plus	138	*	733

*not available on the LSI-11
**not measured

Oregon Software has found similar size and speed rankings of these language processors using other algorithms including a matrix inversion and the Whetstone Benchmarks.

Additional benchmark data is available from:

Oregon Software

2340 S.W. Canyon Road
Portland, Oregon 97201
Telephone: 503/226-7760
TWX 910-464-4779

ER : O.S. LOGO

ALL BACKGROUND SOLIDS TO BE PMS 293 BLUE.

PANTONE
293C

3

Oregon
Software

3½"